/\*

Navicat Premium Data Transfer

Source Server : localhost

Source Server Type : MySQL

Source Server Version : 50624

Source Host : localhost

Source Database : sqlexam

Target Server Type : MySQL

Target Server Version : 50624

File Encoding : utf-8

Date: 10/21/2016 06:46:46 AM

\*/

SET NAMES utf8;

SET FOREIGN\_KEY\_CHECKS = 0;

-- ----------------------------

-- Table structure for `class`

-- ----------------------------

DROP TABLE IF EXISTS `class`;

CREATE TABLE `class` (

`cid` int(11) NOT NULL AUTO\_INCREMENT,

`caption` varchar(32) NOT NULL,

PRIMARY KEY (`cid`)

) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Records of `class`

-- ----------------------------

BEGIN;

INSERT INTO `class` VALUES ('1', '三年二班'), ('2', '三年三班'), ('3', '一年二班'), ('4', '二年九班');

COMMIT;

-- ----------------------------

-- Table structure for `course`

-- ----------------------------

DROP TABLE IF EXISTS `course`;

CREATE TABLE `course` (

`cid` int(11) NOT NULL AUTO\_INCREMENT,

`cname` varchar(32) NOT NULL,

`teacher\_id` int(11) NOT NULL,

PRIMARY KEY (`cid`),

KEY `fk\_course\_teacher` (`teacher\_id`),

CONSTRAINT `fk\_course\_teacher` FOREIGN KEY (`teacher\_id`) REFERENCES `teacher` (`tid`)

) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Records of `course`

-- ----------------------------

BEGIN;

INSERT INTO `course` VALUES ('1', '生物', '1'), ('2', '物理', '2'), ('3', '体育', '3'), ('4', '美术', '2');

COMMIT;

-- ----------------------------

-- Table structure for `score`

-- ----------------------------

DROP TABLE IF EXISTS `score`;

CREATE TABLE `score` (

`sid` int(11) NOT NULL AUTO\_INCREMENT,

`student\_id` int(11) NOT NULL,

`course\_id` int(11) NOT NULL,

`num` int(11) NOT NULL,

PRIMARY KEY (`sid`),

KEY `fk\_score\_student` (`student\_id`),

KEY `fk\_score\_course` (`course\_id`),

CONSTRAINT `fk\_score\_course` FOREIGN KEY (`course\_id`) REFERENCES `course` (`cid`),

CONSTRAINT `fk\_score\_student` FOREIGN KEY (`student\_id`) REFERENCES `student` (`sid`)

) ENGINE=InnoDB AUTO\_INCREMENT=53 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Records of `score`

-- ----------------------------

BEGIN;

INSERT INTO `score` VALUES ('1', '1', '1', '10'), ('2', '1', '2', '9'), ('5', '1', '4', '66'), ('6', '2', '1', '8'), ('8', '2', '3', '68'), ('9', '2', '4', '99'), ('10', '3', '1', '77'), ('11', '3', '2', '66'), ('12', '3', '3', '87'), ('13', '3', '4', '99'), ('14', '4', '1', '79'), ('15', '4', '2', '11'), ('16', '4', '3', '67'), ('17', '4', '4', '100'), ('18', '5', '1', '79'), ('19', '5', '2', '11'), ('20', '5', '3', '67'), ('21', '5', '4', '100'), ('22', '6', '1', '9'), ('23', '6', '2', '100'), ('24', '6', '3', '67'), ('25', '6', '4', '100'), ('26', '7', '1', '9'), ('27', '7', '2', '100'), ('28', '7', '3', '67'), ('29', '7', '4', '88'), ('30', '8', '1', '9'), ('31', '8', '2', '100'), ('32', '8', '3', '67'), ('33', '8', '4', '88'), ('34', '9', '1', '91'), ('35', '9', '2', '88'), ('36', '9', '3', '67'), ('37', '9', '4', '22'), ('38', '10', '1', '90'), ('39', '10', '2', '77'), ('40', '10', '3', '43'), ('41', '10', '4', '87'), ('42', '11', '1', '90'), ('43', '11', '2', '77'), ('44', '11', '3', '43'), ('45', '11', '4', '87'), ('46', '12', '1', '90'), ('47', '12', '2', '77'), ('48', '12', '3', '43'), ('49', '12', '4', '87'), ('52', '13', '3', '87');

COMMIT;

-- ----------------------------

-- Table structure for `student`

-- ----------------------------

DROP TABLE IF EXISTS `student`;

CREATE TABLE `student` (

`sid` int(11) NOT NULL AUTO\_INCREMENT,

`gender` char(1) NOT NULL,

`class\_id` int(11) NOT NULL,

`sname` varchar(32) NOT NULL,

PRIMARY KEY (`sid`),

KEY `fk\_class` (`class\_id`),

CONSTRAINT `fk\_class` FOREIGN KEY (`class\_id`) REFERENCES `class` (`cid`)

) ENGINE=InnoDB AUTO\_INCREMENT=17 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Records of `student`

-- ----------------------------

BEGIN;

INSERT INTO `student` VALUES ('1', '男', '1', '理解'), ('2', '女', '1', '钢蛋'), ('3', '男', '1', '张三'), ('4', '男', '1', '张一'), ('5', '女', '1', '张二'), ('6', '男', '1', '张四'), ('7', '女', '2', '铁锤'), ('8', '男', '2', '李三'), ('9', '男', '2', '李一'), ('10', '女', '2', '李二'), ('11', '男', '2', '李四'), ('12', '女', '3', '如花'), ('13', '男', '3', '刘三'), ('14', '男', '3', '刘一'), ('15', '女', '3', '刘二'), ('16', '男', '3', '刘四');

COMMIT;

-- ----------------------------

-- Table structure for `teacher`

-- ----------------------------

DROP TABLE IF EXISTS `teacher`;

CREATE TABLE `teacher` (

`tid` int(11) NOT NULL AUTO\_INCREMENT,

`tname` varchar(32) NOT NULL,

PRIMARY KEY (`tid`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8;

-- ----------------------------

-- Records of `teacher`

-- ----------------------------

BEGIN;

INSERT INTO `teacher` VALUES ('1', '张磊老师'), ('2', '李平老师'), ('3', '刘海燕老师'), ('4', '朱云海老师'), ('5', '李杰老师');

COMMIT;

SET FOREIGN\_KEY\_CHECKS = 1;

表结构和数据

2、查询“生物”课程比“物理”课程成绩高的所有学生的学号；

思路：

    获取所有有生物课程的人（学号，成绩） - 临时表

    获取所有有物理课程的人（学号，成绩） - 临时表

    根据【学号】连接两个临时表：

        学号  物理成绩   生物成绩

    然后再进行筛选

        select A.student\_id,sw,ty from

        (select student\_id,num as sw from score left join course on score.course\_id = course.cid where course.cname = '生物') as A

        left join

        (select student\_id,num  as ty from score left join course on score.course\_id = course.cid where course.cname = '体育') as B

        on A.student\_id = B.student\_id where sw > if(isnull(ty),0,ty);

3、查询平均成绩大于60分的同学的学号和平均成绩；

    思路：

        根据学生分组，使用avg获取平均值，通过having对avg进行筛选

        select student\_id,avg(num) from score group by student\_id having avg(num) > 60

4、查询所有同学的学号、姓名、选课数、总成绩；

    select score.student\_id,sum(score.num),count(score.student\_id),student.sname

    from

        score left join student on score.student\_id = student.sid

    group by score.student\_id

5、查询姓“李”的老师的个数；

    select count(tid) from teacher where tname like '李%'

    select count(1) from (select tid from teacher where tname like '李%') as B

6、查询没学过“叶平”老师课的同学的学号、姓名；

    思路：

        先查到“李平老师”老师教的所有课ID

        获取选过课的所有学生ID

        学生表中筛选

    select \* from student where sid not in (

        select DISTINCT student\_id from score where score.course\_id in (

            select cid from course left join teacher on course.teacher\_id = teacher.tid where tname = '李平老师'

        )

    )

7、查询学过“001”并且也学过编号“002”课程的同学的学号、姓名；

    思路：

        先查到既选择001又选择002课程的所有同学

        根据学生进行分组，如果学生数量等于2表示，两门均已选择

    select student\_id,sname from

    (select student\_id,course\_id from score where course\_id = 1 or course\_id = 2) as B

    left join student on B.student\_id = student.sid group by student\_id HAVING count(student\_id) > 1

8、查询学过“叶平”老师所教的所有课的同学的学号、姓名；

    同上，只不过将001和002变成 in (叶平老师的所有课)

9、查询课程编号“002”的成绩比课程编号“001”课程低的所有同学的学号、姓名；

    同第1题

10、查询有课程成绩小于60分的同学的学号、姓名；

    select sid,sname from student where sid in (

        select distinct student\_id from score where num < 60

    )

11、查询没有学全所有课的同学的学号、姓名；

    思路：

        在分数表中根据学生进行分组，获取每一个学生选课数量

        如果数量 == 总课程数量，表示已经选择了所有课程

        select student\_id,sname

        from score left join student on score.student\_id = student.sid

        group by student\_id HAVING count(course\_id) = (select count(1) from course)

12、查询至少有一门课与学号为“001”的同学所学相同的同学的学号和姓名；

    思路：

        获取 001 同学选择的所有课程

        获取课程在其中的所有人以及所有课程

        根据学生筛选，获取所有学生信息

        再与学生表连接，获取姓名

        select student\_id,sname, count(course\_id)

        from score left join student on score.student\_id = student.sid

        where student\_id != 1 and course\_id in (select course\_id from score where student\_id = 1) group by student\_id

13、查询至少学过学号为“001”同学所有课的其他同学学号和姓名；

        先找到和001的学过的所有人

        然后个数 ＝ 001所有学科     ＝＝》 其他人可能选择的更多

        select student\_id,sname, count(course\_id)

        from score left join student on score.student\_id = student.sid

        where student\_id != 1 and course\_id in (select course\_id from score where student\_id = 1) group by student\_id having count(course\_id) ＝ (select count(course\_id) from score where student\_id = 1)

14、查询和“002”号的同学学习的课程完全相同的其他同学学号和姓名；

        个数相同

        002学过的也学过

        select student\_id,sname from score left join student on score.student\_id = student.sid where student\_id in (

            select student\_id from score  where student\_id != 1 group by student\_id HAVING count(course\_id) = (select count(1) from score where student\_id = 1)

        ) and course\_id in (select course\_id from score where student\_id = 1) group by student\_id HAVING count(course\_id) = (select count(1) from score where student\_id = 1)

15、删除学习“叶平”老师课的score表记录；

    delete from score where course\_id in (

        select cid from course left join teacher on course.teacher\_id = teacher.tid where teacher.name = '叶平'

    )

16、向SC表中插入一些记录，这些记录要求符合以下条件：①没有上过编号“002”课程的同学学号；②插入“002”号课程的平均成绩；

    思路：

        由于insert 支持

                inset into tb1(xx,xx) select x1,x2 from tb2;

        所有，获取所有没上过002课的所有人，获取002的平均成绩

    insert into score(student\_id, course\_id, num) select sid,2,(select avg(num) from score where course\_id = 2)

    from student where sid not in (

        select student\_id from score where course\_id = 2

    )

17、按平均成绩从低到高 显示所有学生的“语文”、“数学”、“英语”三门的课程成绩，按如下形式显示： 学生ID,语文,数学,英语,有效课程数,有效平均分；

    select sc.student\_id,

        (select num from score left join course on score.course\_id = course.cid where course.cname = "生物" and score.student\_id=sc.student\_id) as sy,

        (select num from score left join course on score.course\_id = course.cid where course.cname = "物理" and score.student\_id=sc.student\_id) as wl,

        (select num from score left join course on score.course\_id = course.cid where course.cname = "体育" and score.student\_id=sc.student\_id) as ty,

        count(sc.course\_id),

        avg(sc.num)

    from score as sc

    group by student\_id desc

18、查询各科成绩最高和最低的分：以如下形式显示：课程ID，最高分，最低分；

    select course\_id, max(num) as max\_num, min(num) as min\_num from score group by course\_id;

19、按各科平均成绩从低到高和及格率的百分数从高到低顺序；

    思路：case when .. then

    select course\_id, avg(num) as avgnum,sum(case when score.num > 60 then 1 else 0 END)/count(1)\*100 as percent from score group by course\_id order by avgnum asc,percent desc;

20、课程平均分从高到低显示（现实任课老师）；

    select avg(if(isnull(score.num),0,score.num)),teacher.tname from course

    left join score on course.cid = score.course\_id

    left join teacher on course.teacher\_id = teacher.tid

    group by score.course\_id

21、查询各科成绩前三名的记录:(不考虑成绩并列情况)

    select score.sid,score.course\_id,score.num,T.first\_num,T.second\_num from score left join

    (

    select

        sid,

        (select num from score as s2 where s2.course\_id = s1.course\_id order by num desc limit 0,1) as first\_num,

        (select num from score as s2 where s2.course\_id = s1.course\_id order by num desc limit 3,1) as second\_num

    from

        score as s1

    ) as T

    on score.sid =T.sid

    where score.num <= T.first\_num and score.num >= T.second\_num

22、查询每门课程被选修的学生数；

    select course\_id, count(1) from score group by course\_id;

23、查询出只选修了一门课程的全部学生的学号和姓名；

    select student.sid, student.sname, count(1) from score

    left join student on score.student\_id  = student.sid

     group by course\_id having count(1) = 1

24、查询男生、女生的人数；

    select \* from

    (select count(1) as man from student where gender='男') as A ,

    (select count(1) as feman from student where gender='女') as B

25、查询姓“张”的学生名单；

    select sname from student where sname like '张%';

26、查询同名同姓学生名单，并统计同名人数；

    select sname,count(1) as count from student group by sname;

27、查询每门课程的平均成绩，结果按平均成绩升序排列，平均成绩相同时，按课程号降序排列；

    select course\_id,avg(if(isnull(num), 0 ,num)) as avg from score group by course\_id order by avg     asc,course\_id desc;

28、查询平均成绩大于85的所有学生的学号、姓名和平均成绩；

    select student\_id,sname, avg(if(isnull(num), 0 ,num)) from score left join student on score.student\_id = student.sid group by student\_id;

29、查询课程名称为“数学”，且分数低于60的学生姓名和分数；

    select student.sname,score.num from score

    left join course on score.course\_id = course.cid

    left join student on score.student\_id = student.sid

    where score.num < 60 and course.cname = '生物'

30、查询课程编号为003且课程成绩在80分以上的学生的学号和姓名；

    select \* from score where score.student\_id = 3 and score.num > 80

31、求选了课程的学生人数

    select count(distinct student\_id) from score

    select count(c) from (

        select count(student\_id) as c from score group by student\_id) as A

32、查询选修“杨艳”老师所授课程的学生中，成绩最高的学生姓名及其成绩；

    select sname,num from score

    left join student on score.student\_id = student.sid

    where score.course\_id in (select course.cid from course left join teacher on course.teacher\_id = teacher.tid where tname='张磊老师') order by num desc limit 1;

33、查询各个课程及相应的选修人数；

    select course.cname,count(1) from score

    left join course on score.course\_id = course.cid

    group by course\_id;

34、查询不同课程但成绩相同的学生的学号、课程号、学生成绩；

    select DISTINCT s1.course\_id,s2.course\_id,s1.num,s2.num from score as s1, score as s2 where s1.num = s2.num and s1.course\_id != s2.course\_id;

35、查询每门课程成绩最好的前两名；

    select score.sid,score.course\_id,score.num,T.first\_num,T.second\_num from score left join

    (

    select

        sid,

        (select num from score as s2 where s2.course\_id = s1.course\_id order by num desc limit 0,1) as first\_num,

        (select num from score as s2 where s2.course\_id = s1.course\_id order by num desc limit 1,1) as second\_num

    from

        score as s1

    ) as T

    on score.sid =T.sid

    where score.num <= T.first\_num and score.num >= T.second\_num

36、检索至少选修两门课程的学生学号；

    select student\_id from score group by student\_id having count(student\_id) > 1

37、查询全部学生都选修的课程的课程号和课程名；

    select course\_id,count(1) from score group by course\_id having count(1) = (select count(1) from student);

38、查询没学过“叶平”老师讲授的任一门课程的学生姓名；

    select student\_id,student.sname from score

    left join student on score.student\_id = student.sid

    where score.course\_id not in (

        select cid from course left join teacher on course.teacher\_id = teacher.tid where tname = '张磊老师'

    )

    group by student\_id

39、查询两门以上不及格课程的同学的学号及其平均成绩；

    select student\_id,count(1) from score where num < 60 group by student\_id having count(1) > 2

40、检索“004”课程分数小于60，按分数降序排列的同学学号；

    select student\_id from score where num< 60 and course\_id = 4 order by num desc;

41、删除“002”同学的“001”课程的成绩；

    delete from score where course\_id = 1 and student\_id = 2